5-2015

ADVISER: A web-based algorithm portfolio deviser

Mustafa MISIR
Singapore Management University, mustafamisir@smu.edu.sg

Stephanus Daniel HANDOKO
Singapore Management University, dhandoko@smu.edu.sg

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

DOI: https://doi.org/10.1007/978-3-319-19084-6_3

Citation

MISIR, Mustafa; HANDOKO, Stephanus Daniel; and LAU, Hoong Chuin. ADVISER: A web-based algorithm portfolio deviser. (2015). *Learning and Intelligent Optimization: 9th International Conference, LION 9, Lille, France, January 12-15, 2015. Revised Selected Papers*. 23-28. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/2793

ADVISER: A Web-based Algorithm Portfolio Deviser

Mustafa Mısır, Stephanus Daniel Handoko, Hoong Chuin Lau

School of Information Systems, Singapore Management University, Singapore
{mustafamisir, dhandoko, hclau}@smu.edu.sg

1 Introduction

The basic idea of algorithm portfolio [1] is to create a mixture of diverse algorithms that complement each other's strength so as to solve a diverse set of problem instances. Algorithm portfolios have taken on a new and practical meaning today with the wide availability of multi-core processors: from an enterprise perspective, the interest is to make best use of parallel machines within the organization by running different algorithms simultaneously on different cores to solve a given problem instance. Parallel execution of a portfolio of algorithms as suggested by [2,3] a number of years ago has thus become a practical computing paradigm.

However, algorithm portfolios to date has remained largely a research pursuit among algorithm designers. For algorithm portfolios to become truly usable by enterprises, we need to enable an end-user to easily obtain an algorithm portfolio when he/she provides a raw set of algorithms and has at his/her disposal a K-core machine. This raises an interesting research challenge: given n target algorithms—some parameter-less and some parameterized—as well as a reference set of problem instances (hereinafter will be referred to as the training instances), how do we automatically construct an algorithm portfolio with a maximum size of K such that together the algorithms in the portfolio are capable of solving the problem instances in the reference set effectively when executed in parallel? Our goal is to generate a portfolio of $k \leq K$ algorithms that are sufficiently diverse from each other and altogether solve the reference instances effectively.

Several software libraries or frameworks have been already introduced in the literature. Hydra [4] is a tuning-based portfolio building strategy that allows incorporating existing parameter tuning and algorithm portfolio techniques. ISAC [5] constructs parameter tuning-based portfolios via instance clustering. SufTra [6] employ problem-independent features to perform instance-specific tuning. LLAMA[1] [7] is an algorithm portfolio selection toolkit implemented in R. HyFlex[2] [8] is a hyper-heuristic framework with iterative heuristic selection methods to solve optimisation problems in a problem-independent manner. All of these frameworks to our knowledge are targeted for use by algorithm developers and not for an end-user in mind.

[1] `https://bitbucket.org/lkotthoff/llama`
[2] `http://www.hyflex.org/`

We present in this paper the ADVISER, an automated Algorithm portfolio DeVISER service that combines ideas from algorithm configuration [9], algorithm selection [10], and portfolio generation within a single framework. To maximize usability by an end-user, ADVISER is a web interface system. Providing such a system over the web is inspired from a another web-based platform dedicated to algorithm configuration, called AutoParTune [3] [6].

The remaining of this paper is organized as follows. Section 2 describes the proposed ADVISER in greater detail. Section 3 presents the success of parallel portfolio recommended by ADVISER through a use-case. Section 4 briefly describes our web-based system. Finally, Section 5 concludes this work and presents directions for future works.

2 ADVISER

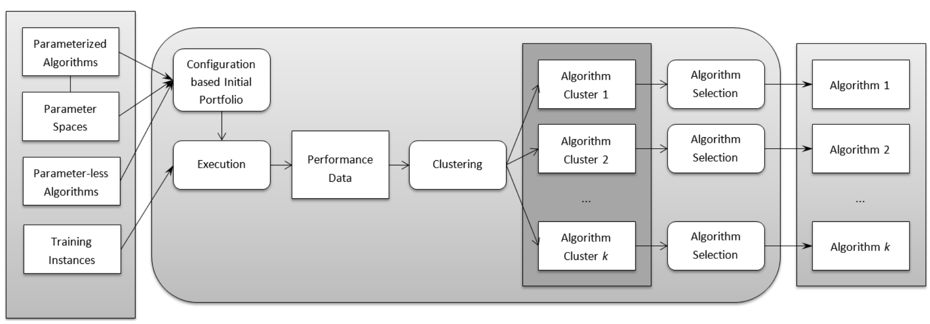

Fig. 1. Workflow of ADVISER

Fig. 1 summarizes the workflow of ADVISER through block diagram. Given a mixture of n parameter-less and parameterized target algorithms as well as a set of training instances as the input, ADVISER first performs *algorithm configuration* and *algorithm selection* to generate a portfolio of $k \leq K$ (configured) algorithms as the output. Parameter-less algorithms directly gets included in the initial portfolio, whereas ADVISER performs algorithm configuration (such as applying ParamILS [11], F-Race [12] and Post-Selection [13]) for each parameterized algorithm to determine the best configuration to be included in the initial portfolio. Performance data is then obtained by executing all algorithms in the initial portfolio on the training instances. Performance data of an algorithm when it runs on an instance refers to a number representing solution quality. The algorithms in the initial portfolio are then clustered based on their performance data and the time taken to achieve such performance. A simple k-means clustering is used for this purpose. Finally, a representative algorithm is chosen

[3] `http://research.larc.smu.edu.sg/autopartune/`

from each cluster via algorithm selection. In this work, we consider choosing the single best algorithm in each cluster for simplicity, where "single best" refers to the algorithm which performs best among the other algorithms in the cluster on most training instances.

3 Case Study

In the following, we present results with $K = 4$ for two parametric algorithms on the Quadratic Assignment Problem (QAP). The first is a population-based a memetic algorithm (MA), and the second is a single-point simulated annealing-tabu search (SA-TS) [14] hybrid metaheuristic.

Table 1. Configuration spaces of MA and SA-TS

Method	Type	Name	Range
MA	Categorical	Crossover (C)	[0, 1, 2, 3, 4]
	Continuous	Mutation Rate (M)	[0, 1]
	Categorical	Local Search (L)	[0, 1, 2, 3]
SA-TS	Integer	Initial Temperature (T)	[4000, 6500]
	Continuous	Cooling Factor (C)	[0.85, 0.95]
	Integer	Tabu List Length (L)	[5, 10]

Table 1 shows the algorithms and their configuration spaces. Both algorithms have three parameters to be set. MA has two categorial parameters and one integer parameter. These categorical parameters are used to represent which crossover and local search operators are to be used while the integer parameter indicates the mutation level. The upper bound values of these categorical parameters refer to the cases where no operator of that type is applied. The two parameters of SA-TS including initial temperature and cooling factor, are for simulated annealing. For the tabu search part, only an integer parameter specifying the tabu list length needs to be set.

Table 2. Portfolio suggested by ADVISER for the QAP using MA and SA-TS

Method	Configuration
MA	-C 4 -M 0.4 -L 2
SA-TS	-T 6500 -C 0.9 -L 5

Table 2 shows the resulting portfolio constructed by using 20 QAP instances. The portfolio is composed of MA and SA-TS with one configuration each ($k =$ 2) instead of four ($K = 4$) since ADVISER detected that there is no need to run that many configurations in parallel. Since the single best algorithm-configuration pair is selected from each cluster, the overall single best which is the MA configuration, automatically is a part of the portfolio.

The portfolio of MA and SA-TS is then tested on 42 QAP instances. The results revealed that MA with the given configuration finds superior results on 28 instances while SA-TS outperforms MA on 12 instances. Both algorithms deliver the same quality solutions on the remaining 2 instances. In other words, the diversity expected from the portfolio is achieved and delivered 12 better solutions compared to the configured single best algorithm, i.e. MA.

4 Web Interface

The ADVISER web interface, shown in Figure 2, is available via the following link: `http://research.larc.smu.edu.sg/adviser/`. A user needs to specify some training instances and the target algorithms as the inputs. The user will then receive an email with the instructions to verify his/her request. After verification, a process involving *algorithm configuration* and *algorithm selection* described in Section 2 is started to build the portfolio. Once the process is completed, the user will receive a notification email along with the portfolio of $k \leq K$ (configured) algorithms as the output.

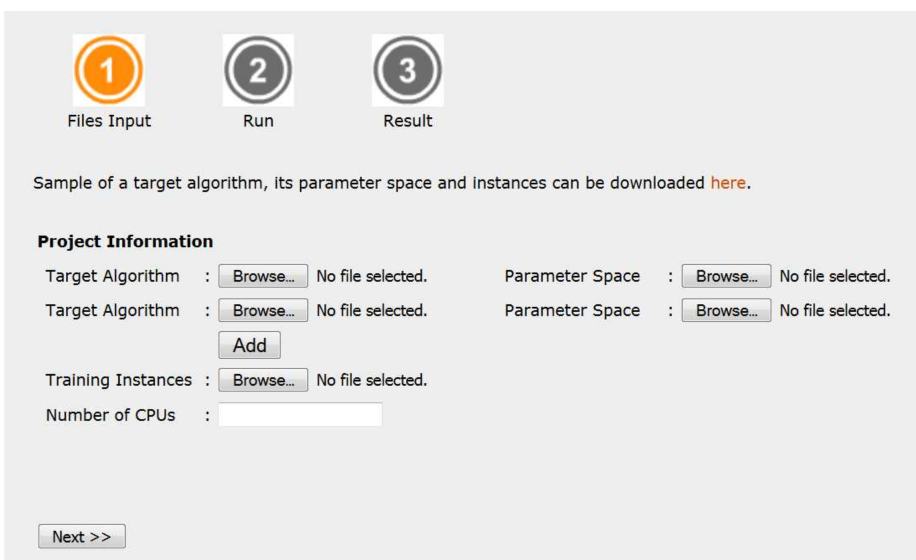

Fig. 2. ADVISER web interface

Each target algorithm should be provided in .exe which can be run as follows. After calling an algorithm, it should return a value representing the quality of the resulting solution.

```
algorithm.exe -I instance_file -S seed ... OtherParameters
```

Alongside with each parametric algorithm, a parameter space file should be given in the following form. In a parameter space file, for each parameter, there should be a parameter name (e.g. `INITIAL_TEMPERATURE`), a parameter argument (e.g. `"-T"`), parameter type information (`i`: integer, `r`: continuous, `c`: categorical) and the range of values (lower and upper bounds for integer and continuous parameters) to be set.

```
INITIAL_TEMPERATURE "-T" i [4000, 6500]
```

ADVISER has been developed in Java. In addition to the presented system, a number of existing parameter tuning related components are integrated. Among those components, a Design of Experiments (DOE) [15] implementation is used to reduce the initial parameter configuration space of each parametric algorithm. SufTra [6] is incorporated for determining similar instances in order to fasten a training process by using a small yet representative instance set. Post-Selection [13] is embedded as a parameter tuner.

5 Conclusion

We believe ADVISER is the first step towards unifying the concepts of algorithm configuration, selection, and portfolio generation with an end-user in mind. The workflow of ADVISER shows how the three components play different yet interrelated roles. Moving forward, we hope to incorporate various techniques of algorithm configuration and selection and allow some degrees of customizations. Options to use instance or algorithmic features, whenever available, will also be explored.

References

1. Huberman, B., Lukose, R., Hogg, T.: An economics approach to hard computational problems. Science **275**(5296) (1997) 51
2. Gomes, C., Selman, B.: Algorithm portfolios. Artificial Intelligence **126**(1) (2001) 43–62
3. Petrik, M., Zilberstein, S.: Learning parallel portfolios of algorithms. Annals of Mathematics and Artificial Intelligence **48**(1) (2006) 85–106
4. Xu, L., Hoos, H., Leyton-Brown, K.: Hydra: Automatically configuring algorithms for portfolio-based selection. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10). (2010) 210–216
5. Kadioglu, S., Malitsky, Y., Sellmann, M., Tierney, K.: ISAC–instance-specific algorithm configuration. In: Proceedings of the 19th European Conference on Artificial Intelligence (ECAI'10). (2010) 751–756

6. Lindawati, Yuan, Z., Lau, H., Zhu, F.: Automated parameter tuning framework for heterogeneous and large instances: Case study in quadratic assignment problem. In Pardalos, P., Nicosia, G., eds.: Proceedings of the 6th Learning and Intelligent OptimizatioN Conference (LION'13). LNCS (2013)
7. Kotthoff, L.: LLAMA: leveraging learning to automatically manage algorithms. Technical Report arXiv:1306.1031 (2013)
8. Ochoa, G., Hyde, M., Curtois, T., Vazquez-Rodriguez, J., Walker, J., Gendreau, M., Kendall, G., McCollum, B., Parkes, A., Petrovic, S., Burke, E.: Hyflex: A benchmark framework for cross-domain heuristic search. In: European Conference on Evolutionary Computation in Combinatorial Optimisation(EvoCOP'12). Volume 7245 of LNCS., Berlin, Springer (2012) 136–147
9. Hutter, F., Hoos, H., Stutzle, T.: Automatic algorithm configuration based on local search. In: Proceedings of the National Conference on Artificial Intelligence. Volume 22., Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; (2007) 1152
10. Rice, J.: The algorithm selection problem. Advances in Computers **15** (1976) 65–118
11. Hutter, F., Hoos, H., Leyton-Brown, K., Stützle, T.: ParamILS: an automatic algorithm configuration framework. Journal of Artificial Intelligence Research **36**(1) (2009) 267–306
12. Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T.: F-race and iterated f-race: An overview. Experimental methods for the analysis of optimization algorithms (2010) 311–336
13. Yuan, Z., Stützle, T., Montes de Oca, M.A., Lau, H.C., Birattari, M.: An analysis of post-selection in automatic configuration. In: Proceeding of the 15th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO'13), ACM (2013) 1557–1564
14. Ng, K., Gunawan, A., Poh, K.: A hybrid algorithm for the quadratic assignment problem. In: Proceedings of international Conference on Scientific Computing, Nevada, USA (July 14–17 2008)
15. Gunawan, A., Lau, H.C., Lindawati: Fine-tuning algorithm parameters using the design of experiments approach. In: the 5th Learning and Intelligent OptimizatioN Conference (LION'11). Volume 6683 of LNCS. Springer (2011) 278–292